

1/5/1 (Item 1 from file: 351)  
DIALOG(R) File 351:Derwent WPI  
(c) 2003 Thomson Derwent. All rts. reserv.

012425673 \*\*Image available\*\*

WPI Acc No: 1999-231781/199920

XRPX Acc No: N99-171734

External data storage apparatus

Patent Assignee: SONY CORP (SONY )

Inventor: FUSE H; SASSA A.

Number of Countries: 031 Number of Patents: 010

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
EP 910020	A1	19990421	EP 98118439	A	19980929	199920 B
AU 9887138	A	19990422	AU 9887138	A	19980929	199927
JP 11110141	A	19990423	JP 97267177	A	19970930	199927
CN 1220424	A	19990623	CN 98125817	A	19980930	199943
SG 71848	A1	20000418	SG 983896	A	19980925	200027
KR 99030334	A	19990426	KR 9841922	A	19980930	200028
US 6330634	B1	20011211	US 98162332	A	19980928	200204
EP 910020	B1	20021120	EP 98118439	A	19980929	200277
AU 753121	B	20021010	AU 9887138	A	19980929	200279
DE 69809527	E	20030102	DE 609527	A	19980929	200310
			EP 98118439	A	19980929	

Priority Applications (No Type Date): JP 97267177 A 19970930

Patent Details:

Patent No. Kind Lan Pg Main IPC Filing Notes

EP 910020 A1 E 30 G06F-011/14

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT  
LI LT LU LV MC MK NL PT RO SE SI

AU 9887138 A G06F-009/445

JP 11110141 A 23 G06F-003/06

CN 1220424 A G06F-012/00

SG 71848 A1 G06F-003/06

KR 99030334 A G06F-012/00

US 6330634 B1 G06F-012/00

EP 910020 B1 E G06F-011/14

Designated States (Regional): DE FR NL

AU 753121 B G06F-009/445 Previous Publ. patent AU 9887138

DE 69809527 E G06F-011/14 Based on patent EP 910020

Abstract (Basic): EP 910020 A1

NOVELTY - Boot data is stored in a number of different blocks, and an identification number indicating whether boot data in each block in which boot data has been stored, is new or old is stored in each of the blocks.

DETAILED DESCRIPTION - Data is erased in units of predetermined blocks, and one block incorporates a block in which boot data which is first read when the external storage apparatus is booted up and stored. Boot data is stored in each of the blocks in which boot data has been stored. An identification number indicating whether boot data stored in each block in which boot data has been stored is old or new, is stored in each of the blocks. An INDEPENDENT CLAIM is included for; a data processing method with which boot data is read when an external memory, arranged to erase data in units of predetermined blocks, is booted up and stored in the external memory.

USE - External storage unit with function to erase data of predetermined blocks, and processing data stored in external storage.

ADVANTAGE - Prevents occurrence of error when boot data is read.

DESCRIPTION OF DRAWING(S) - The drawing shows the structure of a memory card to which the invention is applied.

External storage apparatus (2)

Controller (11)

Flash memory unit (12)

Interface sequencer (13)

Flash memory interface sequencer (14)

Page buffer (15)  
Error correction circuit (16)  
Command generator (17)  
pp; 30 DwgNo 2/12

Title Terms: EXTERNAL; DATA; STORAGE; APPARATUS

Derwent Class: T01; U21

International Patent Class (Main): G06F-003/06; G06F-009/445; G06F-011/14;  
G06F-012/00

International Patent Class (Additional): G06F-001/24; G06F-003/08;  
G06F-012/08; G11C-016/00; G11C-016/02

File Segment: EPI

1/5/2 (Item 1 from file: 347)

DIALOG(R) File 347:JAPIO

(c) 2003 JPO & JAPIO. All rts. reserv.

06168594 \*\*\*Image available\*\*

## EXTERNAL STORAGE DEVICE AND DATA PROCESSING METHOD

PUB. NO.: 11-110141 A]  
PUBLISHED: April 23, 1999 (19990423)

INVENTOR(s): FUSE HIROAKI  
SASA SATORU

APPLICANT(s): SONY CORP.

APPL. NO.: 09-267177 [JP 97267177]

FILED: September 30, 1997 (19970930)

INTL CLASS: G06F-003/06; G06F-003/08; G11C-016/02

### ABSTRACT

PROBLEM TO BE SOLVED: To provide an external storage device, hardly generating an error at the time of boot data read and high in reliability, by providing plural different blocks, in which boot data are respectively stored, and providing an identification number storage part for storing an identification number showing whether the stored boot data are new or old.

SOLUTION: The plural different blocks are provided for respectively storing the boot data. Namely, the storage area of a flash memory is divided into plural blocks to be the units of data erasure. In this case, in these blocks, there are boot blocks for storing the boot data of data to be first read by a data processor 1 when a memory card 2 is started and data blocks for writing arbitrary data. Then, each block storing the boot data is equipped with the identification number storage part for storing the identification number showing whether the boot data stored in that block are new or old.

COPYRIGHT: (C)1999, JPO

(19)日本国特許庁 (JP)

## (12) 公開特許公報 (A)

(11)特許出願公開番号

特開平11-110141

(43)公開日 平成11年(1999)4月23日

(51) Int.Cl.<sup>6</sup>  
 G 06 F 3/06 305  
 3/08  
 G 11 C 16/02

識別記号  
305

F I  
 G 06 F 3/06 305 A  
 3/08 C  
 G 11 C 17/00 601 U

Corresponding to  
KR 1999-030334

審査請求 有 請求項の数 6 OL (全 23 頁)

(21)出願番号 特願平9-267177

(22)出願日 平成9年(1997)9月30日

(71)出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72)発明者 布施 博明  
東京都品川区北品川6丁目7番35号 ソニ  
ー株式会社内(72)発明者 佐々 哲  
東京都品川区北品川6丁目7番35号 ソニ  
ー株式会社内

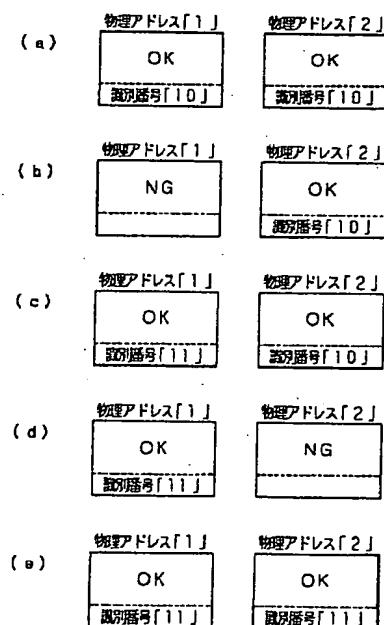
(74)代理人 弁理士 小池 晃 (外2名)

(54)【発明の名称】 外部記憶装置及びデータ処理方法

## (57)【要約】

【課題】 データの消去を所定のブロック単位で行うフラッシュメモリを備えた外部記憶装置について、ブートデータ読み出し時にエラーが発生しないようにする。

【解決手段】 複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納する。そして、起動時に、上記識別番号に基づいて、複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動する。また、起動時に、上記識別番号に基づいて、複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換える。



## 【特許請求の範囲】

【請求項1】 データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置であって、複数の異なるブロックにブートデータをそれぞれ格納するとともに、

ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とする外部記憶装置。

【請求項2】 起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータが読み出され、当該ブートデータを用いて起動されることを特徴とする請求項1記載の外部記憶装置。

【請求項3】 起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧が判別され、古いブートデータがある場合には当該ブートデータが最新のブートデータに書き換えることを特徴とする請求項1記載の外部記憶装置。

【請求項4】 データの消去を所定のブロック単位で行う外部記憶装置に、当該外部記憶装置の起動時に最初に読み出されるブートデータを格納する際に、複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とするデータ処理方法。

【請求項5】 上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動することを特徴とする請求項4記載のデータ処理方法。

【請求項6】 上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換えることを特徴とする請求項4記載のデータ処理方法。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】 本発明は、データの消去が所定のブロック単位で行われる記憶手段を備えた外部記憶装置、並びにそのような外部記憶装置に格納するデータについてのデータ処理方法に関する。

## 【0002】

【従来の技術】 パーソナルコンピュータやデジタルスチルカメラ等のようなデータ処理装置に用いられる外部記憶装置として、いわゆるフラッシュメモリを備えた外部記憶装置がある。

【0003】 フラッシュメモリを備えた外部記憶装置は、記憶領域を複数のブロックに分割し、データ領域の管理をブロック単位で行う。すなわち、例えば、データの消去を行う際は、ブロック単位で行う。また、例えば、記憶領域内に回復不能なエラーが生じたときには、当該エラーが生じた領域を含むブロックを使用しないようにする。なお、以下の説明では、このようなブロックのことを使用不可ブロックと称する。

【0004】 そして、このような外部記憶装置では、当該外部記憶装置にアクセスするために必要な情報等が含まれるブートデータを、予め所定のブロックに格納しておく。そして、当該外部記憶装置を起動する際は、最初にブートデータを読み出し、その後、このブートデータに基づいて当該外部記憶装置へのアクセスを行い、データの書き込みや読み出しを行う。

【0005】 このような外部記憶装置において、ブートデータには、通常、当該外部記憶装置へのアクセスに不可欠な情報が含まれており、ブートデータが読み出せなくなってしまうと、当該外部記憶装置へのアクセスが不可能となってしまう。したがって、ブートデータには、非常に高い信頼性が要求される。

## 【0006】

【発明が解決しようとする課題】 従来、ブートデータが格納されるブロック（以下、ブートブロックと称する。）は、予め設定された所定位置のブロックとされていた。そして、通常使用するブートブロックが使用不可ブロックとなってしまっても、ブートデータを読み出せるように、ブートデータをコピーしたものを他のブロックにも格納しておくようになっていた。そして、従来の外部記憶装置では、信頼性をより高めるために、ブートデータのコピーを多数用意し、それらをそれぞれ異なるブロックに格納しておくようになっていた。しかしながら、ブートデータのコピーを多数用意しておく手法は、記憶容量の有効利用の点では好ましくない。

【0007】 また、従来は、ブートデータが更新されたときに、ブートデータのコピー全てが正しく最新のブートデータに更新されたかを判別するようにはしていなかった。したがって、何らかの利用により、ブートデータのコピーが正しく最新のブートデータに更新されなければ、最新の正しいブートデータを読み出せなくなってしまう、外部記憶装置に対する正常なアクセスができないくなってしまうような場合があった。

【0008】 また、従来は、ブートデータのコピーを用意していたとしても、それらのコピーは、オリジナルのブートデータの読み出しに失敗したときに使用されるだけであった。したがって、従来は、オリジナルのブートデータが格納されたブートブロックの一部のビットが反転してしまったりして、オリジナルのブートデータの内容が正しくなくなっていたとしても、当該ブートデータを読み出すことさえできれば、当該ブートデータがその

まま使用されてしまうという問題があった。

【0009】本発明は、以上のような従来の実情に鑑みて提案されたものであり、記憶領域を有効に利用しながらも、ブートデータ読み出し時にエラーが発生し難く、信頼性の高い外部記憶装置を提供することを目的としている。また、本発明は、そのような外部記憶装置を実現するデータ処理方法を提供することも目的としている。

【0010】

【課題を解決するための手段】本発明に係る外部記憶装置は、データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置である。そして、複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴としている。

【0011】この外部記憶装置は、例えば、起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータが読み出され、当該ブートデータを用いて起動される。

【0012】また、この外部記憶装置は、例えば、起動時に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧が判別され、古いブートデータがある場合には当該ブートデータが最新のブートデータに書き換えられる。

【0013】一方、本発明に係るデータ処理方法は、データの消去を所定のブロック単位で行う外部記憶装置に、当該外部記憶装置の起動時に最初に読み出されるブートデータを格納する際に、複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とする。

【0014】このデータ処理方法では、例えば、上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動する。

【0015】また、このデータ処理方法では、例えば、上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換える。

【0016】

【発明の実施の形態】以下、本発明の実施の形態について、図面を参考しながら詳細に説明する。

【0017】本発明が適用されるシステムの一例について、その全体構成を図1に示す。このシステムは、ホスト側システムとなるデータ処理装置1と、シリアルイン

ターフェースを介してデータ処理装置1に接続される外部記憶装置であるメモリカード2とから構成される。

【0018】なお、ここでは、データ処理装置1とメモリカード2との間でのデータのやり取りをシリアルインターフェースによって行うシステムを例に挙げるが、本発明は、データのやり取りをパラレルインターフェースによって行うシステムに対しても適用可能である。

【0019】データ処理装置1は、演算処理装置(CPU)3と、内部メモリ4と、補助記憶装置5と、シリアルインターフェース回路6とを備え、これらがバス7によって相互に接続されてなる。このデータ処理装置1は、例えば、補助記憶装置5に格納されているプログラムを読み出して、当該プログラムを、内部メモリ4をワークエリアとして使用して、CPU3により実行する。このとき、必要に応じて、シリアルインターフェース回路6を介してメモリカード2との間でデータのやり取りを行う。

【0020】なお、本発明が適用されるシステムに使用されるデータ処理装置は、外部記憶装置との間でデータのやり取りが可能なものであるならば特に限定されるものではなく、本発明は、パーソナルコンピュータ、デジタルスチルカメラ、デジタルビデオカメラ等、種々のデータ処理装置に適用可能である。

【0021】データ処理装置1とメモリカード2とは、シリアルインターフェースによって接続されており、具体的には、少なくとも3本のデータ線SCLK, State, DIOによって接続される。すなわち、データ処理装置1とメモリカード2とは、少なくとも、データ伝送時にクロック信号を伝送する第1のデータ線SCLKと、データ伝送時に必要なステータス信号を伝送する第2のデータ線Stateと、メモリカード2に書き込むデータ又はメモリカード2から読み出すデータ等をシリアルに伝送する第3のデータ線DIOとによって接続され、これらを介して、データ処理装置1とメモリカード2との間でのデータのやり取りを行う。

【0022】データ処理装置1とメモリカード2との間でのデータのやり取りは、通常、ヘッダーと実データとから構成されるファイル単位で行われる。なお、ファイルのヘッダーには、例えば、ファイルにアクセスするための情報や、データ処理装置1で実行されるプログラムで必要とされる情報等が格納される。

【0023】メモリカード2は、図2に示すように、いわゆるコントロールICからなるコントローラ11と、コントローラ11によって管理されるフラッシュメモリ12とを備えている。

【0024】コントローラ11は、シリアル/パラレル変換やパラレル/シリアル変換等を行うシリアル/パラレル・パラレル/シリアル・インターフェース・シーケンサ13(以下、S/P&P/S・インターフェース・シーケンサ13と称する。)と、フラッシュメモリ12

へのインターフェースを司るフラッシュメモリ・インターフェース・シーケンサ14と、S/P&P/S・インターフェース・シーケンサ13とフラッシュメモリ・インターフェース・シーケンサ14との間でやり取りされるデータを一時的に記憶するページバッファ15と、エラー訂正の処理を行うエラー訂正回路16と、フラッシュメモリ12へのアクセスを制御する制御コマンドの生成等を行うコマンドジェネレータ17と、このメモリカード2のバージョン情報や各種属性情報等が格納されているコンフィグレーションROM18と、各回路に対してそれらの動作に必要なクロック信号を供給する発振器19とを備えている。

【0025】S/P&P/S・インターフェース・シーケンサ13は、少なくとも上述した3本のデータ線SCLK, State, DIOを介して、データ処理装置1のシリアルインターフェース回路6に接続され、これらのデータ線SCLK, State, DIOを介して、データ処理装置1との間でデータのやり取りを行う。すなわち、S/P&P/S・インターフェース・シーケンサ13は、ページバッファ15から送られてきたパラレルデータをシリアルデータに変換して、データ処理装置1のシリアルインターフェース回路6へ送出する。また、S/P&P/S・インターフェース・シーケンサ13は、データ処理装置1のシリアルインターフェース回路6から送られてきたシリアルデータをパラレルデータに変換して、ページバッファ15へ送出する。

【0026】このS/P&P/S・インターフェース・シーケンサ13とデータ処理装置1との間でのシリアルデータの伝送は、第1のデータ線SCLKによってデータ処理装置1から送られてくるクロック信号によって同期を取りながら、第3のデータ線DIOによって行われる。このとき、第3のデータ線DIOによってやり取りされるシリアルデータのデータ種別は、第2のデータ線Stateによって伝送されるステータス信号によって判別される。ここで、シリアルデータの種別には、例えば、フラッシュメモリ12に書き込むべきデータ、フラッシュメモリ12から読み出されたデータ、又はこのメモリカード2の動作を制御するための制御データ等がある。なお、ステータス信号は、メモリカード2の状態を示すためにも使用される。ステータス信号によって示されるメモリカード2の状態には、例えば、メモリカード2が何らかの処理の最中でデータ処理装置1からのデータ入力を受け付けない状態や、メモリカード2の側での処理が終了してデータ処理装置1からのデータ入力を待っている状態等がある。

【0027】また、S/P&P/S・インターフェース・シーケンサ13は、データ処理装置1から送られてきたデータがメモリカード2の動作を制御するための制御データである場合には、当該制御データをコマンドジェネレータ17に送出する。

【0028】コマンドジェネレータ17は、データ処理装置1からS/P&P/S・インターフェース・シーケンサ13を介して送られてきた制御データに基づいて、フラッシュメモリ12へのアクセスを制御する制御コマンドを生成し、当該制御コマンドをフラッシュメモリ・インターフェース・シーケンサ14へ送出する。フラッシュメモリ・インターフェース・シーケンサ14は、後述するように、この制御コマンドに基づいて、フラッシュメモリ12にデータを書き込んだり、フラッシュメモリ12からデータを読み出したりする。

【0029】なお、このコマンドジェネレータ17には、誤消去防止スイッチ20が接続されている。そして、この誤消去防止スイッチ20がオンになっているときには、フラッシュメモリ12に書かれているデータを消去するように指示する制御データがデータ処理装置1から送られてきたとしても、コマンドジェネレータ17は、フラッシュメモリ12に書かれているデータを消去するような制御コマンドを生成しない。すなわち、このメモリカード2は、誤消去防止スイッチ20によって、フラッシュメモリ12に保存されているデータの消去が行えない状態と、フラッシュメモリ12に保存されているデータの消去が行える状態とを切り換えることが可能となっている。

【0030】S/P&P/S・インターフェース・シーケンサ13とフラッシュメモリ・インターフェース・シーケンサ14との間に配されたページバッファ15は、いわゆるバッファメモリであり、S/P&P/S・インターフェース・シーケンサ13とフラッシュメモリ・インターフェース・シーケンサ14との間でやり取りされるデータを一時的に記憶する。

【0031】すなわち、S/P&P/S・インターフェース・シーケンサ13からフラッシュメモリ・インターフェース・シーケンサ14へ送られるデータは、先ず、S/P&P/S・インターフェース・シーケンサ13からページバッファ15に送られて、このページバッファ15によって一時的に記憶される。このとき、ページバッファ15に記憶されたデータは、エラー訂正回路16によってエラー訂正符号が付けられる。そして、エラー訂正符号が付けられたデータは、ページバッファ15から所定のページ単位毎（例えば1ページ=512バイトとされる。）に、フラッシュメモリ・インターフェース・シーケンサ14へと送られる。

【0032】或いは、フラッシュメモリ・インターフェース・シーケンサ14からS/P&P/S・インターフェース・シーケンサ13へ送られるデータは、先ず、フラッシュメモリ・インターフェース・シーケンサ14からページバッファ15に送られて、このページバッファ15によって一時的に記憶される。このとき、ページバッファ15に記憶されたデータは、エラー訂正回路16によってエラー訂正処理が施される。そして、エラー訂

正処理が施されたデータは、ページバッファ15から所定のページ単位毎に、S/P&P/S・インターフェース・シーケンサ13へと送られる。

【0033】フラッシュメモリ・インターフェース・シーケンサ14は、コマンドジェネレータ17からの制御コマンドに基づいて、フラッシュメモリ12へのデータの書き込みや、フラッシュメモリ12からのデータの読み出し等を行う。すなわち、フラッシュメモリ・インターフェース・シーケンサ14は、コマンドジェネレータ17からの制御コマンドに基づいて、フラッシュメモリ12からデータを読み出して、当該データを上述のようにページバッファ15を介して、S/P&P/S・インターフェース・シーケンサ13へと送出する。或いは、フラッシュメモリ・インターフェース・シーケンサ14は、コマンドジェネレータ17からの制御コマンドに基づいて、S/P&P/S・インターフェース・シーケンサ13からのデータを、上述のようにページバッファ15を介して受け取り、当該データをフラッシュメモリ12に書き込む。

【0034】コンフィグレーションROM18には、このメモリカード2のバージョン情報や各種属性情報等が格納されている。コンフィグレーションROM18に格納された情報は、必要に応じて、S/P&P/S・インターフェース・シーケンサ13を介してコマンドジェネレータ17によって読み出されて使用される。すなわち、コマンドジェネレータ17は、必要に応じて、コンフィグレーションROM18に格納されている情報を読み出し、この情報に基づいてメモリカード2に関する各種設定を行う。

【0035】以上のようなメモリカード2に対して、フラッシュメモリ12に書き込まれるデータが、上述した3本のデータ線SCLK, State, DIOを介して、データ処理装置1からシリアルデータとして送られてくると、先ず、S/P&P/S・インターフェース・シーケンサ13は、当該シリアルデータをパラレルデータに変換し、当該パラレルデータをページバッファ15へ送出する。ページバッファ15は、S/P&P/S・インターフェース・シーケンサ13から送られてきたデータを一時的に記憶する。このとき、ページバッファ15に記憶されたデータには、エラー訂正回路16によってエラー訂正符号が付けられる。そして、エラー訂正符号が付けられたデータは、所定のページ単位毎にフラッシュメモリ・インターフェース・シーケンサ14に送出される。そして、フラッシュメモリ・インターフェース・シーケンサ14は、ページバッファ15から送られてきたデータを、コマンドジェネレータ17からの制御コマンドに基づいて、フラッシュメモリ12に書き込む。以上の処理により、データ処理装置1から送られてきたデータが、フラッシュメモリ12に書き込まれる。

【0036】また、以上のようなメモリカード2からデ

ータを読み出す際は、先ず、コマンドジェネレータ17からの制御コマンドに基づいて、フラッシュメモリ・インターフェース・シーケンサ14によって、フラッシュメモリ12からデータが読み出される。そして、フラッシュメモリ・インターフェース・シーケンサ14は、フラッシュメモリ12から読み出したデータをページバッファ15に送出する。ページバッファ15は、フラッシュメモリ・インターフェース・シーケンサ14から送られてきたデータを一時的に記憶する。このとき、ページバッファ15に記憶されたデータには、エラー訂正回路16によってエラー訂正処理が施される。そして、エラー訂正処理が施されたデータは、所定のページ単位毎にS/P&P/S・インターフェース・シーケンサ13に送出される。そして、S/P&P/S・インターフェース・シーケンサ13は、ページバッファ15から送られてきたデータを、シリアルデータに変換した上で、上述した3本のデータ線SCLK, State, DIOを介して、データ処理装置1へと送出する。以上の処理により、フラッシュメモリ12から読み出されたデータが、データ処理装置1へと送出される。

【0037】なお、データの書き込みや読み出しを行う際は、フラッシュメモリ12に書き込まれるデータやフラッシュメモリ12から読み出されたデータのやり取りが行われるだけでなく、そのやり取りを制御するための制御データも、データ処理装置1からメモリカード2のS/P&P/S・インターフェース・シーケンサ13へ送られる。この制御データは、S/P&P/S・インターフェース・シーケンサ13からコマンドジェネレータ17に送られる。そして、コマンドジェネレータ17は、S/P&P/S・インターフェース・シーケンサ13から送られてきた制御データに基づいて、フラッシュメモリ12へのアクセスを制御する制御コマンドを生成する。そして、この制御コマンドは、フラッシュメモリ・インターフェース・シーケンサ14に送られ、フラッシュメモリ・インターフェース・シーケンサ14は、この制御コマンドに基づいてフラッシュメモリ12にアクセスして、データの書き込みやデータの読み出しを行う。

【0038】なお、メモリカード2は、上述した3本のデータ線SCLK, State, DIOを備えるだけでなく、その他に、電圧供給用の配線や、通常は使用しないリザーブの配線等を備えていてもよい。例えば、図2並びに後掲する図3では、上述した3本のデータ線SCLK, State, DIOの他に、4本の電源用の配線VSS1, VSS2, VCC, INTと、3本のリザーブの配線RSV1, RSV2, RSV3とをメモリカード2に設けた例を挙げている。

【0039】つぎに、以上のようなメモリカード2の具体的な外形について、図3を参照して説明する。

【0040】メモリカード2は、合成樹脂等からなり平

面形状が長方形とされる薄肉のカード状のケース21に、上述したコントローラ11やフラッシュメモリ12等が内蔵されてなる。そして、このメモリカード2は、当該メモリカード2を装着する装着機構を備えたデータ処理装置1に装着されて使用される。

【0041】このメモリカード2のケース21の前端部には、斜めに切り欠かれた切り欠き部22が形成されており、更に当該切り欠き部22が形成された部分に、10個の凹状部23が形成されている。そして、これらの凹状部23の内部には、メモリカード2がデータ処理装置1の装着装置に装着されたときに、データ処理装置1の接続端子に接続される外部接続用端子が、それぞれ配されている。すなわち、このメモリカード2は、外部接続用端子として10本の端子24a, 24b, 24c, 24d, 24e, 24f, 24g, 24h, 24i, 24jを備えている。これらの外部接続用端子の内訳は、3本のデータ線用の端子24b, 24d, 24h、4本の電源用端子24a, 24f, 24i, 24j、及び3本のリザーブ端子24c, 24e, 24gである。

【0042】また、このメモリカード2のケース21の上面には、誤消去防止部材25が取り付けられている。誤消去防止部材25は、ケース21の内部に収納された上記誤消去防止スイッチ20に係合されており、この誤消去防止部材25をスライド操作することにより、誤消去防止スイッチ20のオン／オフの切り換えを行えるようになっている。

【0043】このメモリカード2には、データ処理装置1の装着装置に装着された際にメモリカード2がデータ処理装置1から脱落しないようにするため、ケース20の側面の一方に円弧状の第1のロック用切欠部26が形成され、ケース20の側面の他方に矩形状の第2のロック用切欠部27が形成されている。そして、このメモリカード2がデータ処理装置1の装着装置に装着されると、メモリカード2が脱落しないように、これらのロック用切欠部26, 27が、データ処理装置1の装着装置に係合される。

【0044】なお、図3に示したメモリカード2は、本発明が適用される外部記憶装置の一例に過ぎない。すなわち、本発明は、外部記憶装置の外形に依存することなく、どんな外形の外部記憶装置にも適用可能である。

【0045】つぎに、以上のようなメモリカード2に搭載されるフラッシュメモリ12の記憶領域の構造について説明する。なお、以下に説明する記憶領域の構造は、本発明が適用される外部記憶装置の記憶領域の構造の一例に過ぎない。

【0046】すなわち、本発明は、データ消去の単位となる複数のブロックに記憶領域が分割されてなるとともに、ブートデータが格納されるブートブロックを備えた外部記憶装置に対して広く適用可能であり、その記憶領域の構造は、以下に説明するような構造でなくてもよ

い。

【0047】このフラッシュメモリ12の記憶領域は、図4(a)に示すように、データ消去の単位となる複数のブロックに分割されてなる。なお、これらのブロックには、このメモリカード2が起動されたときにデータ処理装置1によって最初に読み込まれるデータであるブートデータが格納されるブートブロックと、任意のデータが書き込まれるデータブロックとがある。各ブロックには、それぞれ固有の物理アドレスが付けられている。これらのブロックは、データ消去の単位であると同時に、ファイル管理上の最小単位でもある。すなわち、ファイルは1つ又は複数のブロックに格納され、1つのブロックを複数のファイルで利用することはできない。

【0048】そして、各ブロックは、「1」又は「0」を示す2つの状態を取りうる複数のビットからなり、初期状態では、全てのビットが「1」とされており、ビット単位での変更は「1」から「0」へだけが可能となっている。すなわち、「1」及び「0」からなるデータを書き込む際、「1」については該当するビットをそのまま保持し、「0」については該当するビットを「1」から「0」に変更する。

【0049】そして、一度書き込んだデータを消去する際は、ブロック単位で一括して初期化処理を行い、当該ブロックの全ビットを「1」とする。これにより、当該ブロックに書き込まれたデータが一括して消去され、そのブロックは再びデータの書き込みが可能な状態となる。

【0050】なお、本発明は、上述のように各ビットが2つの状態だけを取りうるフラッシュメモリ（いわゆる2値型のフラッシュメモリ）だけでなく、各ビットが3つ以上の状態を取りうるフラッシュメモリ（いわゆる多値型のフラッシュメモリ）にも適用可能である。

【0051】また、上記フラッシュメモリ12の各ブロックは、図4(b)に示すように、データの書き込みや読み出しの単位となる複数のページから構成される。すなわち、このフラッシュメモリ12にデータを書き込む際は、上述したように、ページ単位にてページバッファ15から送られてきたデータが、フラッシュメモリ・インターフェース・シーケンサ14によってページ単位にてフラッシュメモリ12に書き込まれる。また、このフラッシュメモリ12からデータを読み出す際は、フラッシュメモリ・インターフェース・シーケンサ14によってページ単位毎にデータが読み出されて、ページバッファ15へと送られる。

【0052】各ページは、データエリアと、冗長エリアとを有している。データエリアは、任意のデータが書き込まれる領域である。冗長エリアは、データエリアに書き込まれるデータの管理に必要な情報が格納される領域である。

【0053】具体的には、図4(c)に示すように、ブ

ロックの先頭ページの冗長エリアには、当該ブロックを管理するために必要な情報として、いわゆる分散管理情報が格納される。また、ブロックの2ページ目以降の各ページの冗長エリアにも、予備の分散管理情報として、先頭ページの冗長エリアに格納された分散管理情報と同じものが格納される。ただし、最終ページの冗長エリアには、分散管理情報ではなく、分散管理情報だけでは管理しきれない追加情報として、いわゆる追加管理情報が格納される。

【0054】このように、このフラッシュメモリ12では、各ブロック内の冗長エリアに分散管理情報が格納される。分散管理情報は、当該分散管理情報が格納されたブロックを管理するための情報である。この分散管理情報により、例えば、当該ブロックがファイルの先頭となるブロックであるか否かについての情報や、複数のブロックからファイルが構成される場合にはそれらのブロックの繋がりを示す情報等を得ることができる。なお、この分散管理情報については、後で詳細に説明する。

【0055】そして、このメモリカード2では、各ブロックの分散管理情報を集めることにより、フラッシュメモリ全体を管理するための情報として、いわゆる集合管理情報を作成して、この集合管理情報をファイルとしてフラッシュメモリ12に格納しておくようとする。

【0056】そして、通常は、集合管理情報によって、各ブロックにアクセスするために必要な情報を得るようにする。すなわち、データ処理装置1とメモリカード2との間でデータのやり取りを行う際、データ処理装置1は、集合管理情報をメモリカード2から読み出して内部メモリ4に展開し、この集合管理情報に基づいてメモリカード2にアクセスする。これにより、データアクセスの都度、個々のブロックに格納された分散管理情報をアクセスするような必要がなくなり、より高速なデータアクセスが可能となる。

【0057】つぎに、分散管理情報、追加管理情報、集合管理情報について、更に詳細に説明する。

【0058】分散管理情報は、当該分散管理情報が格納されたブロックを管理するための情報であり、16バイトの冗長エリアに書き込まれてなる。具体的には、図5に示すように、1バイトの可／不可フラグと、1バイトのブロックフラグと、4ビットの最終フラグと、4ビットの参照フラグと、1バイトの管理フラグと、2バイトの論理アドレスと、2バイトの連結アドレスと、3バイトのリザーブ領域と、2バイトの分散管理情報用エラー訂正符号と、3バイトのデータ用エラー訂正符号とからなる。

【0059】可／不可フラグは、ブロックが使用可能状態か使用不可能状態かを示すフラグであり、具体的には、「使用可」と「使用不可」の2つの状態を示す。

「使用可」は、当該ブロックが使用可能な状態を示し、「使用不可」は、当該ブロックが使用不可能な状態であ

ることを示す。例えば、ブロック内に回復不能なエラーが生じたようなときに、この可／不可フラグが「使用不可」に設定され、当該ブロックが使用不可とされる。

【0060】ブロックフラグは、ブロックの状態を示すフラグであり、具体的には、「未使用」「先頭使用」「使用」「未消去」の4つの状態を示す。「未使用」は、当該ブロックが未使用又は消去済みで、初期状態（全ビットが「1」の状態）とされており、直ぐにデータの書き込みが可能な状態を示す。「先頭使用」は、当該ブロックがファイルの先頭で使用されている状態を示す。なお、ブートデータが格納されたブートブロックにおいて、ブロックフラグは「先頭使用」とされる。「使用」は、当該ブロックがファイルの先頭以外で使用されている状態を示す。ブロックフラグが「使用」のとき、当該ブロックは、他のブロックから連結されていることとなる。「未消去」は、当該ブロックに書かれていたデータが不要となった状態を示す。例えば、データの消去を行うときに、取りあえずブロックフラグを「未消去」にしておき、処理時間に余裕があるときに、ブロックフラグが「未消去」になっているブロックを消去するようになる。これにより、消去処理をより効率良く行うことが可能となる。

【0061】最終フラグは、ファイルが終わっているか否かを示すフラグであり、具体的には、「ブロック連続」「ブロック最終」の2つの状態を示す。「ブロック連続」は、次のブロックへの連結があることを示す。すなわち、「ブロック連続」は、当該ブロックに格納されたファイルにはまだ続きがあり、当該ファイルが他のブロックに続いていることを示す。「ブロック最終」は、最終ブロックであることを示す。すなわち、「ブロック最終」は、当該ブロックに格納されたファイルが、このブロックで終了していることを示す。

【0062】参照フラグは、追加管理情報の参照を指定するためのフラグであり、具体的には、「参照情報なし」「参照情報あり」の2つの状態を示す。「参照情報なし」は、ブロックの最終ページの冗長領域に、有効な追加管理情報が存在しないことを示す。「参照情報あり」は、ブロックの最終ページの冗長領域に、有効な追加管理情報が存在していることを示す。

【0063】管理フラグは、ブロックの属性等を示すフラグである。例えば、この管理フラグによって、当該ブロックが読み出し専用ブロックか、或いは書き込みも可能なブロックであるかが示される。また、例えば、この管理フラグによって、当該ブロックがブートブロックであるか、或いはデータブロックであるかが示される。

【0064】論理アドレスは、文字通りそのブロックの論理アドレスを示す。この論理アドレスの値は、データの書き換えを行うときなどに必要に応じて更新される。なお、論理アドレスの値は、正常に処理が行われている限り、同じ論理アドレスの値を同時に複数のブロックが

持つことがないように設定される。

【0065】連結アドレスは、当該ブロックに連結するブロックの論理アドレスを示す。すなわち、ブロックに格納されたファイルにはまだ続きがあり、当該ファイルが他のブロックに続いている場合、連結アドレスには、そのファイルの続きが格納された次のブロックの論理アドレスの値が設定される。

【0066】分散管理情報用エラー訂正符号は、分散管理情報のうち、管理フラグ、論理アドレス、連結アドレス及びリザーブ領域に書き込まれデータを対象としたエラー訂正符号である。なお、可／不可フラグ、ブロックフラグ、最終フラグ及び参照フラグは、分散管理情報用エラー訂正符号によるエラー訂正の対象となっていない。したがって、可／不可フラグ、ブロックフラグ、最終フラグ及び参照フラグは、分散管理情報用エラー訂正符号を更新することなく書き換えることが可能となっている。

【0067】データ用エラー訂正符号は、当該データ用エラー訂正符号が格納されているページのデータエリアに書き込まれたデータを対象としたエラー訂正符号である。

【0068】なお、分散管理情報用エラー訂正符号やデータ用エラー訂正符号は、メモリカード2の内部に配されたエラー訂正回路16によって使用される。したがって、これらのエラー訂正符号を用いてのエラー訂正是、データ処理装置1に依存することなく、メモリカード2に依存した任意の手法を使用することができる。

【0069】追加管理情報は、ブロックの最終ページの16バイトの冗長エリアに格納される情報であり、分散管理情報だけでは管理しきれない追加情報を含んでいる。

【0070】具体的には、追加管理情報は、図6に示すように、1バイトの可／不可フラグと、1バイトのブロックフラグと、4ビットの最終フラグと、4ビットの参照フラグと、1バイトの識別番号と、2バイトの有効データサイズと、5バイトのリザーブ領域と、2バイトの追加管理情報用エラー訂正符号と、3バイトのデータ用エラー訂正符号とからなる。

【0071】ここで、可／不可フラグ、ブロックフラグ、最終フラグ、参照フラグ、リザーブ領域及びデータ用エラー訂正符号については、分散管理情報の場合と同様である。また、追加管理情報用エラー訂正符号は、分散管理情報における分散管理情報用エラー訂正符号に相当するものであり、追加管理情報のうち、識別番号、有効データサイズ及びリザーブ領域に書き込まれデータを対象としたエラー訂正符号である。

【0072】そして、識別番号及び有効データサイズとが、分散管理情報だけでは管理しきれない追加情報として、追加管理情報に含まれている。

【0073】識別番号は、エラー処理用の情報であり、

ブロックのデータを書き換える度に、この識別番号の値がインクリメントされる。この識別番号は、何らかのエラーが発生して、同じ論理アドレスを持つブロックが複数存在するようになってしまった場合に、それらのブロックに書き込まれたデータの新旧を識別するために使用される。なお、識別番号には1バイトの領域が使用され、その値の範囲は「0」から「255」までであり、その初期値は「0」とされる。なお、識別番号が「255」を越えたときには、「0」に戻される。そして、同じ論理アドレスを持つデータブロックが複数存在する場合には、この識別番号の値が小さい方のデータブロックを有効とする。また、後で詳細に説明するように、同じ論理アドレスを持つブートブロックが複数存在する場合には、この識別番号の値が大きい方のブートブロックを有効とする。

【0074】また、有効データサイズは、ブロック内の有効なデータのサイズを示す。すなわち、当該ブロックのデータエリアに空きがある場合、有効データサイズには、当該データエリアに書き込まれたデータのサイズを示す値が設定される。このとき、分散管理情報の参照フラグは「参照情報あり」に設定される。なお、ブロックのデータエリアに空きがない場合、有効データサイズには、当該データエリアに空きがないことを示す値として、「0xffff」が設定される。

【0075】なお、以上のような分散管理情報及び追加管理情報は、ブロック内のデータが更新される毎に、常に最新情報となるように更新される。

【0076】集合管理情報は、上述したように、各ブロックの分散管理情報を集めて作成されてなる情報であり、ファイルとしてフラッシュメモリ12に格納される。すなわち、図7に示すように、各ブロックの分散管理情報から、全ブロックをまとめて管理するための情報である集合管理情報のファイルが作成され、この集合管理情報が所定のブロックのデータエリアに格納される。なお、集合管理情報は、1つのブロックに格納されるものであっても、複数のブロックにわたって格納されるものであってもよい。そして、データ処理装置1は、通常は、この集合管理情報によって、各ブロックにアクセスするために必要な情報を得るようにする。

【0077】このメモリカード2の起動時には、先ず、データ処理装置1によって、ブートブロックからブートデータが読み込まれるが、このブートデータには、集合管理情報が格納されているブロックの物理アドレスが指定されている。そこで、データ処理装置1は、当該物理アドレスに対応したブロックに格納されている集合管理情報を読み出して内部メモリ4に展開し、この集合管理情報に基づいてメモリカード2にアクセスする。なお、フラッシュメモリ12から集合管理情報を正常に読み出せなかった場合、データ処理装置1は、全てのブロックの分散管理情報を読み込み、集合管理情報を再構築して

使用する。

【0078】そして、データ処理装置1は、データの書き換え等を行う毎に、内部メモリ4に展開した集合管理情報を、フラッシュメモリ12の実際の状態と整合するようす（すなわち、分散管理情報の内容と整合するようす）、隨時更新していく。一方、フラッシュメモリ12にファイルとして格納されている集合管理情報は、データの書き換え等を行う毎に更新するのではなく、適当なタイミングにて、その変更内容が一括して更新される。なお、ここでの適当なタイミングとは、例えば、電源を落とす前や、メモリカード2へのアクセスが所定時間以上なされなかったときや、データの書き換えが所定回数以上行われたときなどである。

【0079】一般にフラッシュメモリ12の書き換え可能回数には上限があるが、このようにフラッシュメモリ12にファイルとして格納されている集合管理情報の書き換えをある程度まとめて一括して行うようになると、集合管理情報が格納されているブロックの書き換え回数を削減することができ、メモリカード2の長寿命化を図ることができる。

【0080】つぎに、本発明のポイントとなるブートブロックの取り扱いについて詳細に説明する。

【0081】ブートデータは、メモリカード2を起動したときにデータ処理装置1によって最初に読み込まれるデータであり、ブートデータには、メモリカード2にアクセスするために必要不可欠な情報が含まれている。したがって、ブートデータが格納されたブートブロックにエラーが生じて、ブートデータを正常に読み出すことができなくなると、メモリカード2へのアクセスが不可能になってしまふ。したがって、ブートブロックには、非常に高い信頼性が要求される。

【0082】そこで、本発明を適用したメモリカード2では、常にブートデータをフラッシュメモリ12の先頭の有効な2ブロックにそれぞれ格納しておくようとする。なお、「有効なブロック」とは、使用可能状態にあるブロックのことである。換言すれば、「有効なブロック」とは、可／不可フラグが「使用可」に設定されているブロックである。すなわち、ブートデータをブロックに格納する際に、有効でないブロック（すなわち、可／不可フラグが「使用不可」に設定されているブロック）は、使用することなく飛ばして、有効なブロックにブートデータを格納してブートブロックとする。

【0083】このように、本発明を適用したメモリカード2では、固定された特定のブロックをブートブロックにするのではなく、有効でないブロックが生じたときには、ブートデータを格納するブロックを変更して、有効な2つのブロックに常にブートデータが格納されるようになる。すなわち、例えば、メモリカード2の先頭のブロックにエラーが生じたとしても、常にブートデータは有効なブロックに二重に保持される。したがって、本発

明を適用したメモリカード2の信頼性は非常に高いものとなる。

【0084】また、本発明を適用したメモリカード2では、追加管理情報に含まれている識別番号を、データブロックの場合とブートブロックの場合とで区別して使用する。

【0085】すなわち、データブロックの場合には、同じ論理アドレスのブロックが複数あったときに、それらのブロックに格納されているデータの新旧を識別するために識別番号を使用する。ここで、同じ論理アドレスを持つデータブロックが複数存在するのは、多くの場合、データブロック更新時に何らかのエラーが発生したためであり、このとき、識別番号の値が小さい方のデータブロックに格納されているデータが更新前のデータである。そこで、同じ論理アドレスを持つデータブロックが複数存在する場合には、データを更新する前の状態に戻すために、識別番号の値が小さい方のデータブロックを選択し、当該データブロックに格納されているデータを有効なデータとして採用する。

【0086】なお、データブロックの識別番号は、データブロックに対して論理アドレスが新規に割り当てられたときに初期化されて「0」とされる。また、この識別番号は、データブロックの更新時に1インクリメントされる。なお、上述したように、識別番号が「255」を越えた場合には「0」に戻される。

【0087】一方、ブートブロックの場合には、上述したようにフラッシュメモリ12の先頭の有効な2ブロックに格納されたブートデータについて、それらの新旧を識別するために識別番号を使用する。ブートデータは、メモリカード2へのアクセスに必要不可欠な情報であり、常に最新の情報を用いる必要がある。そして、2つのブートブロックの識別番号が異なるのは、多くの場合、ブートブロックの更新時に何らかのエラーが発生したためであり、このとき、識別番号の値が大きいブートブロックに格納されているブートデータが、より新しい情報を含むブートデータである。そこで、2つのブートブロックについて、それらの識別番号が異なる場合には、識別番号の値が大きい方のブートブロックを選択し、当該ブートブロックに格納されているデータをブートデータとして採用する。

【0088】なお、このメモリカード2は、最初に使用するときにフォーマットされるが、このフォーマット時に、ブートブロックの識別番号は初期化され「0」とされる。そして、この識別番号は、ブートブロックに格納されているブートデータを更新した時に1インクリメントされる。なお、ブートブロックの場合も、データブロックのときと同様に、識別番号が「255」を越えた場合には「0」に戻される。

【0089】つぎに、以上のようにフラッシュメモリ12の先頭の有効な2ブロックをブートブロックにすると

とともに、それらのブートブロックの新旧を識別番号を用いて管理する際の手順について、更に詳細に説明する。

【0090】(1) フォーマット時の手順

上述したように、このメモリカードは、最初に使用するときにフォーマットされる。そして、このフォーマット時にブートブロックが作成される。そこで、まず、フォーマット時のブートブロック作成の手順について説明する。

【0091】フォーマット時に、データ処理装置1は、先ず、メモリカード2のコンフィグレーションROM18に格納されている情報を読み出し、この情報等に基づいてブートブロックに格納するデータ（すなわちブートデータ）を作成する。なお、このデータには、当該データを読み出したときに、そのデータがブートデータであることが分かるように、当該データがブートデータであることを示す情報（以下、ブート識別子と称する。）も含めておく。

【0092】次に、作成したブートデータを、フラッシュメモリ12の先頭の2ブロック（すなわち、物理アドレスの値が最も小さいブロックと、物理アドレスの値がその後に小さいブロック）にそれぞれ書き込み、ブートブロックと予備のブートブロックを作成する。このとき、ブートデータが書き込まれるブロックの追加管理情報の識別番号を「0」に設定する。エラーなく終了すれば、ブートブロックの作成を終了する。

【0093】ここでのブートデータの書き込みは、具体的には、先ず、該当するブロックに対して消去処理を施し、その後、当該ブロックに対してブートデータを書き込むことによって行う。このとき、当該ブロックに対して消去処理が正常に行えなかった場合や、消去処理が施されたブロックに対してブートデータを正常に書き込めなかった場合には、当該ブロックの可／不可フラグが「使用不可」に設定され、当該ブロックは使用不可状態とされる。このとき、可／不可フラグを「使用不可」に設定できなかった場合には、可／不可フラグを「使用不可」に設定する処理を何度もリトライする。それでも設定できない場合には、データ処理装置1はメディア異常と判断し、そのメモリカード2を受け付けないようにする。なお、消去処理が施されたブロックに対してブートデータを正常に書き込めなかった場合には、当該ブロックに途中まで書き込んだブートデータが残ってしまったりするようなことがないように、可／不可フラグを「使用不可」に設定する前に、当該ブロックに対して消去処理を施しておく。

【0094】そして、本発明を適用したメモリカード2では、ブートブロックにしようとしたブロックにエラーが発生した場合には、次に使用可能なブロックに対して、同様にブートデータの書き込みを試みる。そして、正常で且つ内容が同一のブートブロックが2個作成されるまで、この処理を続ける。これにより、内容が同一の

ブートデータがフラッシュメモリ12の先頭の有効な2ブロックにそれぞれ格納され、先頭の有効な2ブロックがブートブロック及び予備のブートブロックとなる。

【0095】ただし、ブートデータの書き込みは、対象となるブロックの物理アドレスが所定の値Mになつたら、内容が同一のブートブロックが2個作成されていなくても、処理を終了する。この所定の値Mは、フラッシュメモリ12の特性等に応じて予め規定しておく。そして、対象となるブロックの物理アドレスが所定の値Mになつたら、ブートブロックが1つだけしか作成されていなくても、ブートブロックの作成を終了する。また、対象となるブロックの物理アドレスが所定の値Mになつても、ブートブロックが1つも作成されない場合には、メディア異常と判断し、そのメモリカード2を受け付けないようにする。

【0096】ところで、一般にフラッシュメモリ12は、書き換え回数が多いブロックでエラーが発生する可能性が高く、一旦エラーが始めると、その後もエラーが生じる確率が高い。そこで、以前ブートブロックとして使用されたが、エラーになって使用不可になったブロックは、たとえそのエラーが一時的なものであつたとしても、そのブロックの再利用はしないようにする。このような規則を採用することにより、非常に重要な情報であるブートデータは、より信頼性の高いブロックに格納されることとなり、ブートブロックの信頼性の向上を図ることができる。なお、この規則は、先頭のブロックから2つめのブートブロックまでの間にだけ、或いは2つめのブートブロックが無い場合には物理アドレスが上記所定の値Mに至るまでの間にだけ適用する。すなわち、この規則は、高い信頼性が要求されるブートブロックについてだけ適用し、それ以降のブロック（すなわちデータブロック）に対しては適用しない。これにより、記憶領域のより有効な利用を図ることができる。

【0097】なお、ブートブロックの読み込み時にエラーが発生した場合、当該ブートブロックを直ぐに使用不可にするようなことはしない。例えば、ブートブロックの書き換え中にメモリカード2がデータ処理装置1から抜かれた場合、ブートブロックの書き換えが中途半端に終了している可能性が高い。このようなとき、そのブートブロックを次に読み込んだ場合にエラーが発生するが、そのブロックは再利用が可能であるので、当該ブロックを使用不可にする必要はない。

【0098】しかしながら、データ処理装置1は、このような読み込みエラーと、フラッシュメモリ12で回復不能な障害が発生したために起きた読み込みエラーとを区別して検出することができない。そこで、ブートブロックの読み込み時にエラーが発生した場合は、当該ブートブロックを消去して改めてブートデータを書き込む。もし、フラッシュメモリ12で回復不能な障害が発生している場合は、ブロック消去やデータ書き込み時にもエ

ラーが発生することとなる。そこで、このようにブロック消去やデータ書き込み時にエラーが生じたときにだけ、そのブロックを使用不可とする。

【0099】(2)ブートブロック更新時の手順  
ブートデータは、常に同じデータとは限らず、メモリカード2の使用に伴い、その内容を変更する必要が生じる。したがって、ブートブロックを更新して、ブートデータを書き換える場合がある。このブートデータの書き換えは、以下に示すルールに基づいて行う。

【0100】・ブートデータの書き換えを行うようなときには、集合管理情報の内容を変更する必要がある。そこで、ブートデータの書き換えを行う前に、フラッシュメモリ1にファイルとして格納されている集合管理情報を無効にする。なお、次に集合管理情報を用いるときは、改めて分散管理情報から構築し直す。

【0101】・ブートデータの内容を書き換える際は、現在ブートデータが格納されているブートブロックに対して消去処理を施し、そのブロックに新たなブートデータを書き込む。すなわち、ブートデータを書き換えるときは、別のブロックを使用することなく、同一ブロックの内容を更新する。ただし、ブートブロックの更新時にエラーが発生し、当該ブロックが使用不可になった場合は、この限りではない。

【0102】・新しいブートデータに書き換える場合は、上述したように2つ作成したブートブロックのうち、物理アドレスが小さい方から行う。このとき、内容を更新したブートブロックの識別番号を1インクリメントする。ただし、書き換え前の識別番号の値が「255」の場合は「0」とする。

【0103】・予備のブートブロック（すなわち物理アドレスの大きいほうのブートブロック）を更新して、もうひとつのブートブロックと同一の内容となるように、新しいブートデータに書き換えたときは、当該予備のブートブロックの識別番号を、もうひとつのブートブロックの識別番号と同じ値にする。

【0104】以上のようなルールに基づいて、ブートブロックを更新する際の手順について、図8を参照して説明する。

【0105】なお、図8では、物理アドレス「1」のブロックが通常使用されるブートブロック（以下、第1のブートブロックと称する。）とされ、物理アドレス「2」のブロックが予備のブートブロック（以下、第2のブートブロックと称する。）とされている例を示している。また、図8では、ブートデータが正常に書き込まれておらず、当該ブートデータの読み出しが可能な状態のことを「OK」として示している。また、ブートデータ書き換えの最中等により、ブートデータの読み出しが不可能になっている状態のことを「NG」として示している。

【0106】図8(a)は、ブートブロック書き換え前

の正常な状態を示している。このとき、第1のブートブロックの識別番号と、第2のブートブロックの識別番号とは、同じ値である。具体的には、図8(a)の例では、第1のブートブロックの識別番号が「10」であり、第2のブートブロックの識別番号も同じ「10」とされている。

【0107】ブートブロックを更新する際は、先ず、図8(b)に示すように、物理アドレスの小さいほうのブートブロック、すなわち第1のブートブロックのほうから、ブートデータの書き換えを行う。このとき、第1のブートブロックは、ブートデータの書き換えの最中であるので、ブートブロックとしては使用できない「NG」状態となる。

【0108】もしもこの段階で、メモリカード2がデータ処理装置1から強制的に抜かれてしまったりして、処理が中断されてしまった場合、再起動時に、第1のブートブロックをブートブロックとして使用することはできない。このときは、予備のブートブロックである第2のブートブロックに格納されているブートデータを読み出してメモリカード2を起動する。また、このときは、第2のブートブロックから読み出されたブートデータに基づいて、第1のブートブロックを再構築する。

【0109】そして、第1のブートブロックが更新されると、図8(c)に示すように、第1のブートブロックの識別番号が1インクリメントされ、本例ではその値が「11」とされる。この段階では、第1のブートブロックには新しいブートデータが格納され、第2のブートブロックには古いブートデータが格納された状態となる。

【0110】したがって、もしもこの段階で、メモリカード2がデータ処理装置1から強制的に抜かれてしまったりして、処理が中断されてしまった場合、再起動時には、第1のブートブロックに格納されている新しいブートデータ（すなわち、識別番号の値が大きいほうのブートブロックに格納されているブートデータ）を読み出してメモリカード2を起動する。また、このときは、識別番号の値が大きいほうのブートブロック（すなわち第1のブートブロック）に格納されているブートデータに基づいて、識別番号の値が小さいほうのブートブロック（すなわち第2のブートブロック）のブートデータを更新する。

【0111】そして、第1のブートブロックの更新が完了したら、次に、図8(d)に示すように、物理アドレスの大きいほうのブートブロック、すなわち第2のブートブロックの更新を行う。このとき、第2のブートブロックは、ブートデータの書き換えの最中であるので、ブートブロックとしては使用できない「NG」状態となる。

【0112】もしもこの段階で、メモリカード2がデータ処理装置1から強制的に抜かれてしまったりして、処理が中断されてしまった場合、再起動時には、第1のブ

ートブロックのブートデータを使用してメモリカード2は起動されるが、予備のブートブロックが存在しない状態となる。したがって、このときは、第1のブートブロックから読み出されたブートデータに基づいて、予備のブートブロックである第2のブートブロックを再構築する。

【0113】そして、第2のブートブロックが更新されると、図8(e)に示すように、第2のブートブロックの識別番号が1インクリメントされ、本例ではその値が「11」とされる。これにより、第1のブートブロックの識別番号と、第2のブートブロックの識別番号とが等しい値となる。以上の処理により、第1のブートブロックにも第2のブートブロックにも新しいブートデータが格納された状態となる。

【0114】(3) メモリカード起動時のブートブロック読み込みの手順

ブートブロックに格納されたブートデータは、メモリカード2がデータ処理装置1に接続され起動されたときに、データ処理装置1に最初に読み込まれる。そこで、つぎに、メモリカード起動時における、データ処理装置1によるブートブロック読み込みの手順について詳細に説明する。

【0115】正常な状態では、同一識別番号を持ち且つ同一内容を持つブートブロックが2つ存在する。そこで、このことを確認する処理を、メモリカード起動時に必ず行う。具体的には、先頭ブロックから順番に、以下に示すような流れで処理を行い、ブートブロックを調べていく。

【0116】・先頭ページの分散管理情報が正常に読めるかを確認する。

【0117】・可/不可フラグが「使用可」になっているかを確認する。

【0118】・ブロックフラグが「先頭使用」になっているかを確認する。

【0119】・ブート識別子を検出して、格納されているデータがブートデータであるかを確認する。

【0120】・最終ページに格納されている追加管理情報が正常に読めるかを確認する。

【0121】・識別番号を読み出す。このとき、1個目のブートブロックの場合には、その値を保存する。2個目のブートブロックの場合には、1個目のブートブロックの識別番号と値が一致するかを確認する。

【0122】・ブートブロック内のデータを読み出す。1個目のブートブロックの場合、読み出したデータは保存しておく。2個目のブートブロックの場合、読み出したデータと保存しておいた1個目のブートブロックのデータとが一致するかを確認する。

【0123】以上のような処理を先頭ブロックから順次行い、同一識別番号を持ち且つ同一内容を持つブートブロックが2つ確認された時点で、ブートブロックの読み

込みを終了する。このように、メモリカード2の起動時に2つのブートブロックの確認を行うことにより、メモリカード2の信頼性を非常に高めることができる。

【0124】なお、物理アドレスが上述した所定の値Mになるまで調べた時点で、ブートブロックが1つしか存在しない場合には、当該ブートブロックに格納されているブートデータを使用してメモリカード2を起動する。このとき、先頭からM番目までのブロックに使用可能なブロックがある場合には、当該ブロックにブートデータを書き込んで、改めて予備のブートブロックを作成する。先頭からM番目までのブロックに使用可能なブロックがない場合には、ブートブロックが1つの状態のままでメモリカード2を動作させる。また、M番目のブロックまで調べた時点で、ブートブロックが存在しなかった場合には、データ処理装置1はメディア異常と判断し、そのメモリカード2を受け付けないようにする。

【0125】なお、2つのブロックにそれぞれブートデータが正常に書かれても識別番号が異なっている場合には、識別番号の大きいほうのブロックを正当なブートブロックとして選択し、メモリカード2の起動には、識別番号の大きいほうのブロックに格納されているブートデータを使用する。ただし、一方の識別番号が「255」で他方の識別番号が「0」の場合には、識別番号が「0」のほうのブロックを正当なブートブロックとして選択し、当該ブロックのブートデータを使用する。また、1つのブロックにだけブートデータが正常に書かれているときには、そのブロックがブートブロックとして使用される。

【0126】つぎに、以上のようなメモリカード起動時のブートブロック読み込みの手順について、図9乃至図12に示すフローチャートを参照して、更に詳細に説明する。なお、ここでは、変数としてI, WB, IDA, IDBを使用する。変数I, WBは、物理アドレスが入力される変数であり、変数IDA, IDBは、識別番号の値が入力される変数である。

【0127】メモリカードの起動時には、図9に示すように、先ず、ステップS1において、変数Iに「0」を代入する。また、変数IDA, IDBに「0」を代入する。次に、ステップS2へ進む。

【0128】ステップS2では、処理の対象となるブロックを、変数Iが示す物理アドレスのブロックとする。次に、ステップS3へ進む。

【0129】ステップS3では、現在処理の対象となっているブロックから、分散管理情報を読み出せるかを判別する。分散管理情報を読み出せれば、ステップS4へ進み、分散管理情報を読み出せなければ、ステップS5へ進む。

【0130】ステップS4では、現在処理の対象となっているブロックの可/不可フラグが「使用可」になっているかを判別する。「使用可」になつていれば、ステッ

アS5へ進み、「使用可」になっていなければ、ステップS22へ進む。

【0131】ステップS5では、現在処理の対象となっているブロックのブロックフラグが「先頭使用」になっているかを判別する。「先頭使用」になっていれば、ステップS6へ進み、「先頭使用」になっていなければ、ステップS24へ進む。

【0132】ステップS6では、現在処理の対象となっているブロックに格納されているデータに、ブート識別子が設定されているかを判別する。すなわち、当該データがブートデータであるかを判別する。ブートデータであれば、ステップS7へ進み、ブートデータでなければ、ステップS28へ進む。

【0133】ステップS7では、現在処理の対象となっているブロックから、追加管理情報を読み出せるかを判別する。追加管理情報を読み出せれば、ステップS8へ進み、追加管理情報を読み出せなければ、ステップS25へ進む。

【0134】ステップS8では、変数IDAに、現在処理の対象となっているブロックの識別番号の値を代入する。次に、ステップS9へ進む。

【0135】ステップS9では、現在処理の対象となっているブロックから、ブートデータを読み出せるかを判別する。ブートデータを読み出せれば、ステップS10へ進み、ブートデータを読み出せなければ、ステップS25へ進む。

【0136】ステップS10では、現在処理の対象となっているブロックからブートデータを読み出し、当該ブートデータを保存する。次に、ステップS11へ進む。

【0137】ステップS11では、変数WBに変数Iの値を代入するとともに、変数Iの値を1インクリメントする。次に、図10のステップS12へ進む。

【0138】ステップS12では、処理の対象となるブロックを、変数Iが示す物理アドレスのブロックとする。次に、ステップS13へ進む。

【0139】ステップS13では、現在処理の対象となっているブロックから、分散管理情報を読み出せるかを判別する。分散管理情報を読み出せれば、ステップS14へ進み、分散管理情報を読み出せなければ、ステップS32へ進む。

【0140】ステップS14では、現在処理の対象となっているブロックの可／不可フラグが「使用可」になっているかを判別する。「使用可」になっていれば、ステップS15へ進み、「使用可」になっていなければ、ステップS29へ進む。

【0141】ステップS15では、現在処理の対象となっているブロックのブロックフラグが「先頭使用」になっているかを判別する。「先頭使用」になっていれば、ステップS16へ進み、「先頭使用」になっていなければ、ステップS31へ進む。

【0142】ステップS16では、現在処理の対象となっているブロックに格納されているデータに、ブート識別子が設定されているかを判別する。すなわち、当該データがブートデータであるかを判別する。ブートデータであれば、ステップS17へ進み、ブートデータでなければ、ステップS35へ進む。

【0143】ステップS17では、現在処理の対象となっているブロックから、追加管理情報を読み出せるかを判別する。追加管理情報を読み出せれば、ステップS18へ進み、追加管理情報を読み出せなければ、ステップS32へ進む。

【0144】ステップS18では、変数IDBに、現在処理の対象となっているブロックの識別番号の値を代入する。次に、ステップS19へ進む。

【0145】ステップS19では、変数IDAの値と変数IDBの値とを比較する。変数IDAの値と変数IDBの値とが等しければ、ステップS20へ進み、変数IDAの値と変数IDBの値とが等しくなければ、ステップS32へ進む。

【0146】ステップS20では、現在処理の対象となっているブロックから、ブートデータを読み出せるかを判別する。ブートデータを読み出せば、ステップS21へ進み、ブートデータを読み出せなければ、ステップS32へ進む。

【0147】ステップS21では、現在処理の対象となっているブロックからブートデータを読み出し、当該ブートデータと、先に読み出して保存しておいたブートデータとが一致しているかを判別する。それらのブートデータが一致しているのは、2つのブートブロックからのブートデータの読み出しが正常に行われたときであるので、これで処理を終了する。一方、それらのブートデータが一致していなければ、ステップS32へ進む。

【0148】一方、図9に示すように、ステップS4で可／不可フラグが「使用可」になっていなかった場合には、上述したようにステップS22へ進む。

【0149】ステップS22では、変数Iの値と、上述した所定の値Mとを比較する。そして、変数Iの値が所定の値Mよりも小さければ、ステップS23へ進む。ステップS22において、変数Iの値が所定の値M以上となるのは、M番目のブロックまで調べてもブートデータが得られなかった場合であり、このときは、エラーとして処理を終了する。

【0150】ステップS23では、変数Iの値を1インクリメントする。そして、ステップS2へ戻って処理を繰り返す。

【0151】また、ステップS5でブロックフラグが「先頭使用」になっていなかった場合は、上述したようにステップS24へ進む。このステップS24では、当該ブロックフラグが「未使用」になっているかを判別する。「未使用」になっていれば、図11のステップS3

6へ進み、「未使用」になっていなければ、ステップ25へ進む。

【0152】ステップS25では、現在処理の対象となっているブロックに対して消去処理を施す。次に、ステップS26へ進む。

【0153】ステップS26では、ステップS25での消去処理が正常に完了したかを判別する。消去処理が正常に完了していれば、図11のステップS36へ進み、消去処理が正常に完了していなければ、ステップ27へ進む。

【0154】ステップS27では、現在処理の対象となっているブロックの可／不可フラグを「使用不可」に設定し、その後、ステップS22へ進んで上述した処理を行う。

【0155】また、ステップS6でデータがブートデータでなかった場合には、上述したようにステップS28へ進む。

【0156】このステップS28では、処理の対象となるブロックを他のブロックに移動する。その後、ステップS25へ進んで上述した処理を行う。

【0157】また、ステップS3で分散管理情報が読み出せなかった場合、ステップS7で追加管理情報を読み出せなかった場合、及びステップS9でブートデータを読み出せなかった場合にも、上述したようにステップS25へ進んで、上述した処理を行う。

【0158】また、図10に示すように、ステップS14で可／不可フラグが「使用可」になっていなかった場合には、上述したようにステップS29へ進む。このステップS29では、変数Iの値と、上述した所定の値Mとを比較する。そして、変数Iの値が所定の値Mよりも小さければ、ステップS30へ進む。一方、ステップS29において変数Iの値が所定の値M以上となるのは、M番目のブロックまで調べても2つめのブートデータが得られなかった場合であり、このときは、ここでブートデータ読み出しの処理を終了し、先に読み出して保存しておいたブートデータを使ってメモリカード2を起動する。

【0159】ステップS30では、変数Iの値を1インクリメントする。そして、ステップS12へ戻って処理を繰り返す。

【0160】また、ステップS15でブロックフラグが「先頭使用」になっていなかった場合は、上述したようにステップ31へ進む。このステップS31では、当該ブロックフラグが「未使用」になっているかを判別する。「未使用」になっていれば、図12のステップS47へ進み、「未使用」になっていなければ、ステップ32へ進む。

【0161】ステップS32では、現在処理の対象となっているブロックに対して消去処理を施す。次に、ステップS33へ進む。

【0162】ステップS33では、ステップS32での消去処理が正常に完了したかを判別する。消去処理が正常に完了していれば、図12のステップS47へ進み、消去処理が正常に完了していなければ、ステップ34へ進む。

【0163】ステップS34では、現在処理の対象となっているブロックの可／不可フラグを「使用不可」に設定し、その後、ステップS29へ進んで上述した処理を行う。

【0164】また、ステップS16でデータがブートデータでなかった場合には、上述したようにステップS35へ進む。このステップS35では、処理の対象となるブロックを他のブロックに移動する。その後、ステップS32へ進んで上述した処理を行う。

【0165】また、ステップS13で分散管理情報が読み出せなかった場合、ステップS17で追加管理情報を読み出せなかった場合、ステップS19で変数IDAの値と変数IDBの値とが一致しなかった場合、ステップS20でブートデータを読み出せなかった場合、及びステップS21でブートデータが一致しなかった場合にも、上述したようにステップS32へ進んで、上述した処理を行う。

【0166】また、図11に示すように、ステップS36では、変数WBに変数Iの値を代入するとともに、変数Iの値を1インクリメントする。次に、ステップS37へ進む。

【0167】ステップS37では、処理の対象となるブロックを、変数Iが示す物理アドレスのブロックとする。次に、ステップS38へ進む。

【0168】ステップS38では、現在処理の対象となっているブロックから、分散管理情報を読み出せるかを判別する。分散管理情報を読み出せれば、ステップS39へ進み、分散管理情報を読み出せなければ、ステップS45へ進む。

【0169】ステップS39では、現在処理の対象となっているブロックの可／不可フラグが「使用可」になっているかを判別する。「使用可」になっていれば、ステップS40へ進み、「使用可」になっていなければ、ステップS45へ進む。

【0170】ステップS40では、現在処理の対象となっているブロックに格納されているデータに、ブート識別子が設定されているかを判別する。すなわち、当該データがブートデータであるかを判別する。ブートデータであれば、ステップS41へ進む。

【0171】ステップS41では、現在処理の対象となっているブロックから、追加管理情報を読み出せるかを判別する。追加管理情報を読み出せれば、ステップS42へ進む。

【0172】ステップS42では、現在処理の対象となっているブロックから、ブートデータを読み出せるかを

判別する。ブートデータを読み出せば、ステップS43へ進む。

【0173】ステップS43では、現在処理の対象となっているブロックからブートデータを読み出し、当該ブートデータを保存する。次に、ステップS44へ進む。

【0174】ステップS44では、変数WBが示す物理アドレスのブロックに、ステップS43で読み出して保存しておいたブートデータを書き込む。このときは、ここでブートデータ読み出しの処理を終了し、ステップS43で読み出して保存しておいたブートデータを使ってメモリカード2を起動する。

【0175】また、ステップS38で分散管理情報が読み出せなかった場合、及びステップS39で可／不可フラグが「使用可」になっていなかった場合には、上述したようにステップS45へ進む。このステップS45では、変数Iの値と、上述した所定の値Mとを比較する。そして、変数Iの値が所定の値Mよりも小さければ、ステップS46へ進む。ステップS45において、変数Iの値が所定の値M以上となるのは、M番目のブロックまで調べてもブートデータが得られなかった場合であり、このときは、エラーとして処理を終了する。

【0176】ステップS46では、変数Iの値を1インクリメントする。そして、ステップS37へ戻って処理を繰り返す。

【0177】また、ステップS40でデータがブートデータではなかった場合、ステップS41で追加管理情報を読み出せなかった場合、及びステップS42でブートデータを読み出せなかった場合には、エラーとして処理を終了する。これは、2つのブートブロックのいずれからもブートデータが読み出せなかった場合である。

【0178】また、図12に示すように、ステップS47では、変数IDBの値が「0xffff」であるかを判別する。変数IDBの値が「0xffff」でなければステップS48へ進み、変数IDBの値が「0xffff」であればステップS49へ進む。

【0179】ステップS48では、変数IDAの値と、変数IDBの値に「1」を加えた値とを比較する。これらの値が等しければ、ステップS49へ進み、等しくなければ、ステップS51へ進む。

【0180】ステップS49では、変数WBに変数Iの値を代入する。次に、ステップS50へ進む。

【0181】ステップS50では、変数WBが示す物理アドレスのブロックに、先に読み出して保存しておいたブートデータを書き込む。このときは、ここでブートデータ読み出しの処理を終了し、先に読み出して保存しておいたブートデータを使ってメモリカード2を起動する。

【0182】ステップS51では、変数IDBの値と、変数IDAの値に「1」を加えた値とを比較する。これらの値が等しければ、ステップS50へ進み上述した処

理を行う。

【0183】一方、ステップS51において、変数IDBの値と、変数IDAの値に「1」を加えた値とが等しくないと判別されるのは、2つのブートブロックの識別番号が、等しくなく且つ連続番号でもない場合である。このような状態となるのは、メモリカード2に何らかの異常があったときである。しかしながら、ブートデータが読み出せなくなっているわけではないので、このときは、例えば手動復旧モードとして、データ処理装置1の側で適切な処理を行うようとする。

【0184】以上がメモリカード起動時のブートブロック読み込みの手順である。このように手順により、メモリカード2の起動時に2つのブートブロックの確認を行うことで、メモリカード2の信頼性を非常に高めることができる。

【0185】

【発明の効果】以上詳細に説明したように、本発明によれば、複数の異なるブロックにブートデータをそれぞれ格納するようにしているので、たとえブートデータが格納されたブロックのうちのいずれかが使用不可能になったとしても、他のブロックのブートデータを用いて、外部記憶装置を起動することができる。

【0186】しかも、それらのブロックにはブートデータの新旧を示す識別番号を格納するようにしているので、常に最新のブートデータを使用するようになることができる。すなわち、本発明によれば、たとえ新しいブートデータと古いブートデータとが混在するような状態になったとしても、常に最新のブートデータを使用するようになることができ、データの一貫性を保証できる。

【0187】したがって、本発明によれば、ブートデータ読み出し時にエラーが発生し難く、信頼性の高い外部記憶装置を提供することができる。

【図面の簡単な説明】

【図1】本発明が適用されるシステムの全体構成を示す図である。

【図2】本発明を適用したメモリカードの構成を示すブロック図である。

【図3】本発明を適用したメモリカードの外観を示す斜視図である。

【図4】本発明を適用したメモリカードの記憶領域の構造を示す図である。

【図5】分散管理情報の構成を示す図である。

【図6】追加管理情報の構成を示す図である。

【図7】各ブロックの分散管理情報から集合管理情報を構築する様子を示す図である。

【図8】ブートブロックを更新する際の手順を示す図である。

【図9】メモリカード起動時のブートブロック読み込みの手順を示すフローチャートである。

【図10】メモリカード起動時のブートブロック読み込

みの手順を示すフローチャートである。

【図11】メモリカード起動時のブートブロック読み込みの手順を示すフローチャートである。

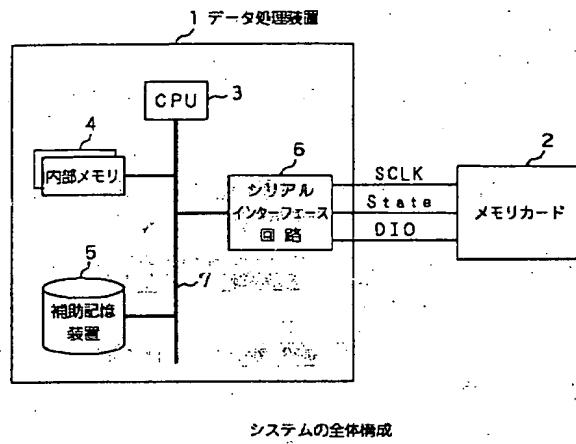
【図12】メモリカード起動時のブートブロック読み込みの手順を示すフローチャートである。

【符号の説明】

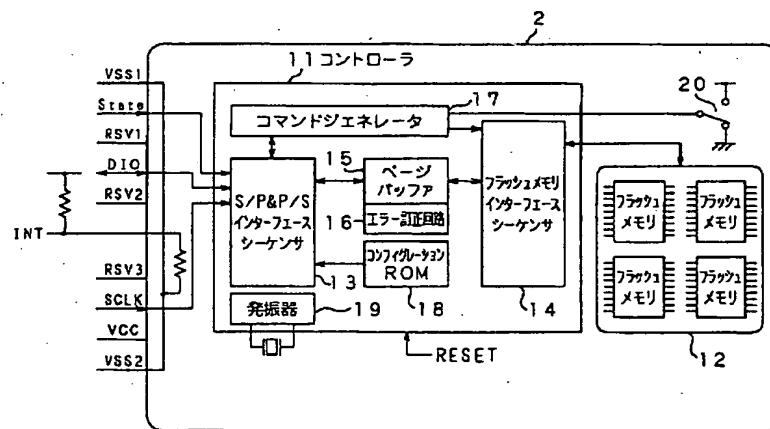
1 データ処理装置、 2 メモリカード、 3 演算処理装置、 4 内部メモリ、 5 補助記憶装置、

6 シリアルインターフェース回路、 7 バス、 1  
1 コントローラ、 12 フラッシュメモリ、 13  
シリアル/パラレル・パラレル/シリアル・インター  
フェース・シーケンサ、 14 フラッシュメモリ・イ  
ンターフェース・シーケンサ、 15 ページバッフ  
ア、 16 エラー訂正回路、 17 コマンドジェネレ  
ータ、 18 コンフィグレーションROM、 19  
発振器、 20 誤消去防止スイッチ

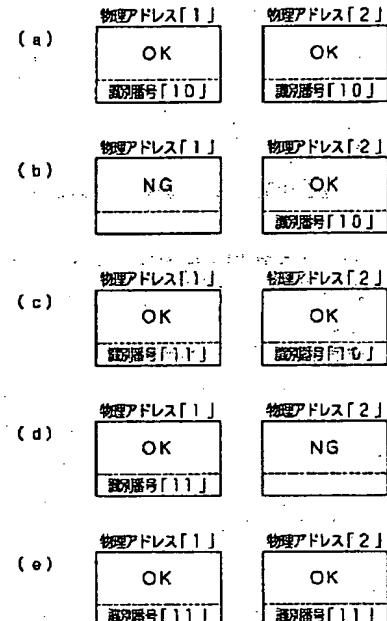
【図1】



【図2】

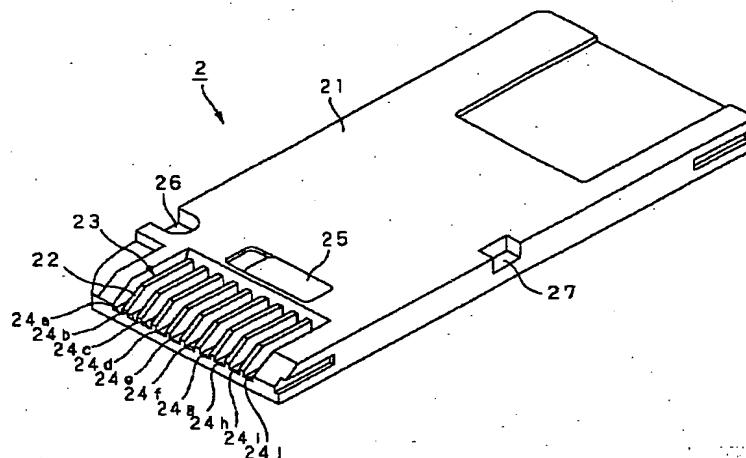


【図8】

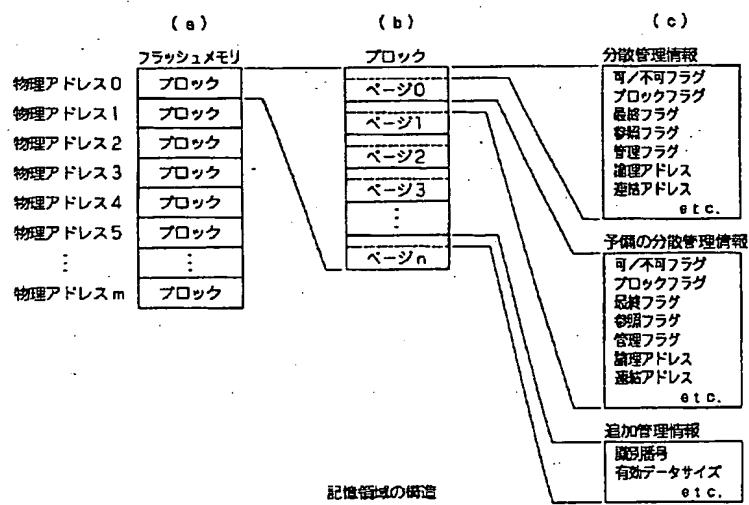


ブートブロックの更新時の手順

〔图3〕



〔図4〕



【図5】

可/不可 フラグ	ロック フラグ	認 識 別 番 号	管理 フラグ	論理アドレス	連結アドレス	リザーブ領域	分散管理情報用 エラー訂正符号	データ用 エラー訂正符号
-------------	------------	-----------------------	-----------	--------	--------	--------	--------------------	-----------------

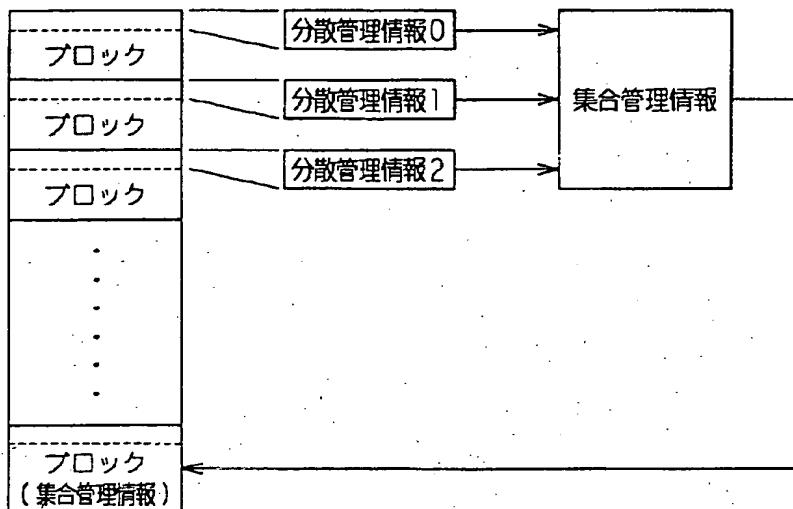
分散管理情報

【図6】

可/不可 フラグ	ロック フラグ	認 識 別 番 号	識別番号	有効データサイズ	リザーブ領域	追加管理情報用 エラー訂正符号	データ用 エラー訂正符号
-------------	------------	-----------------------	------	----------	--------	--------------------	-----------------

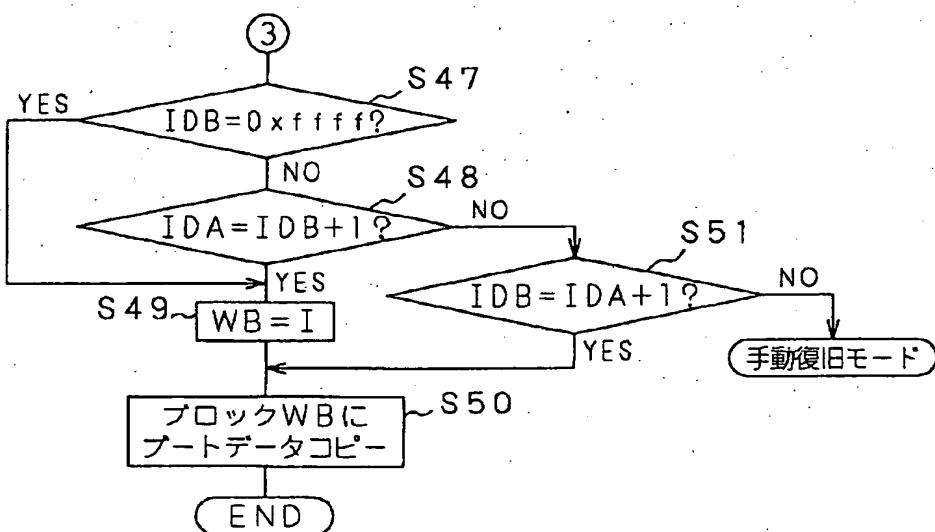
追加管理情報

【図7】



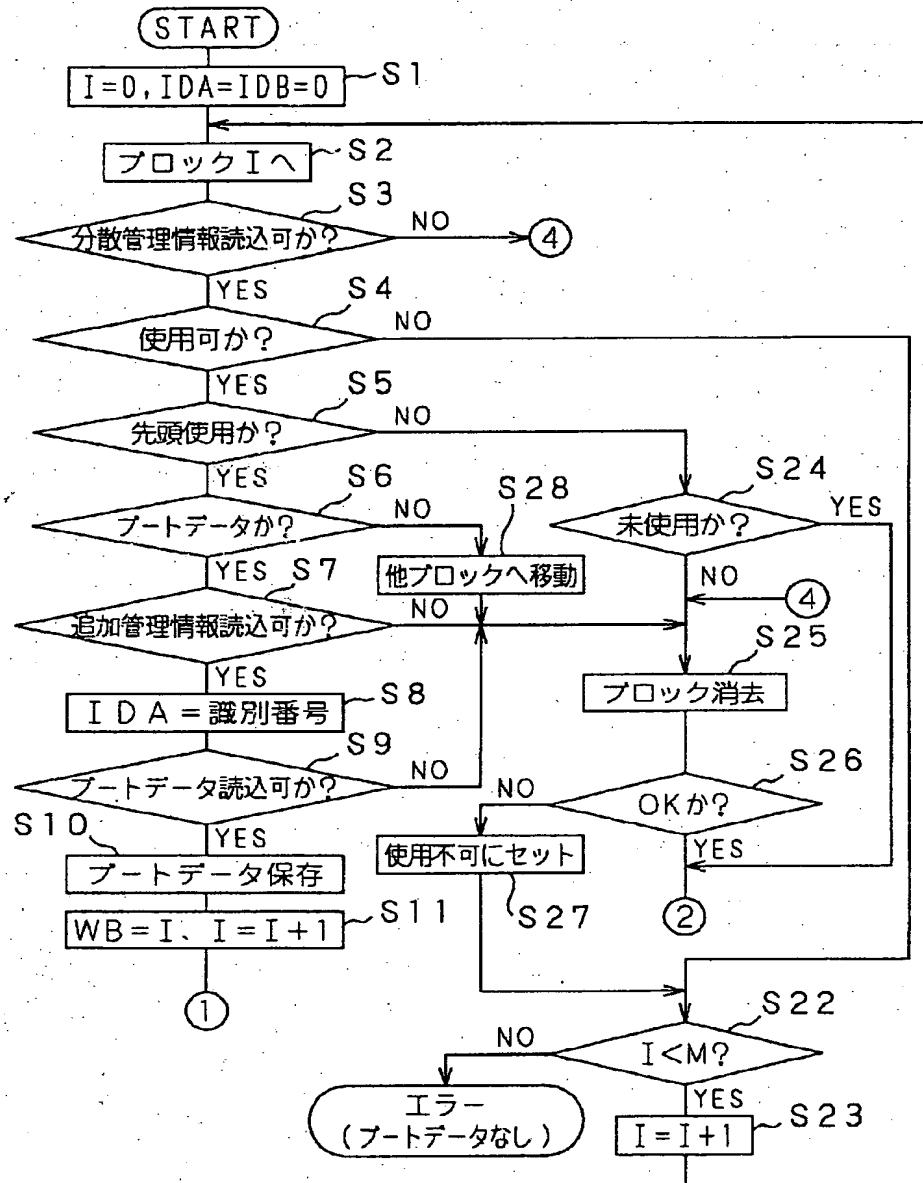
分散管理情報から集合管理情報の構築

【図12】

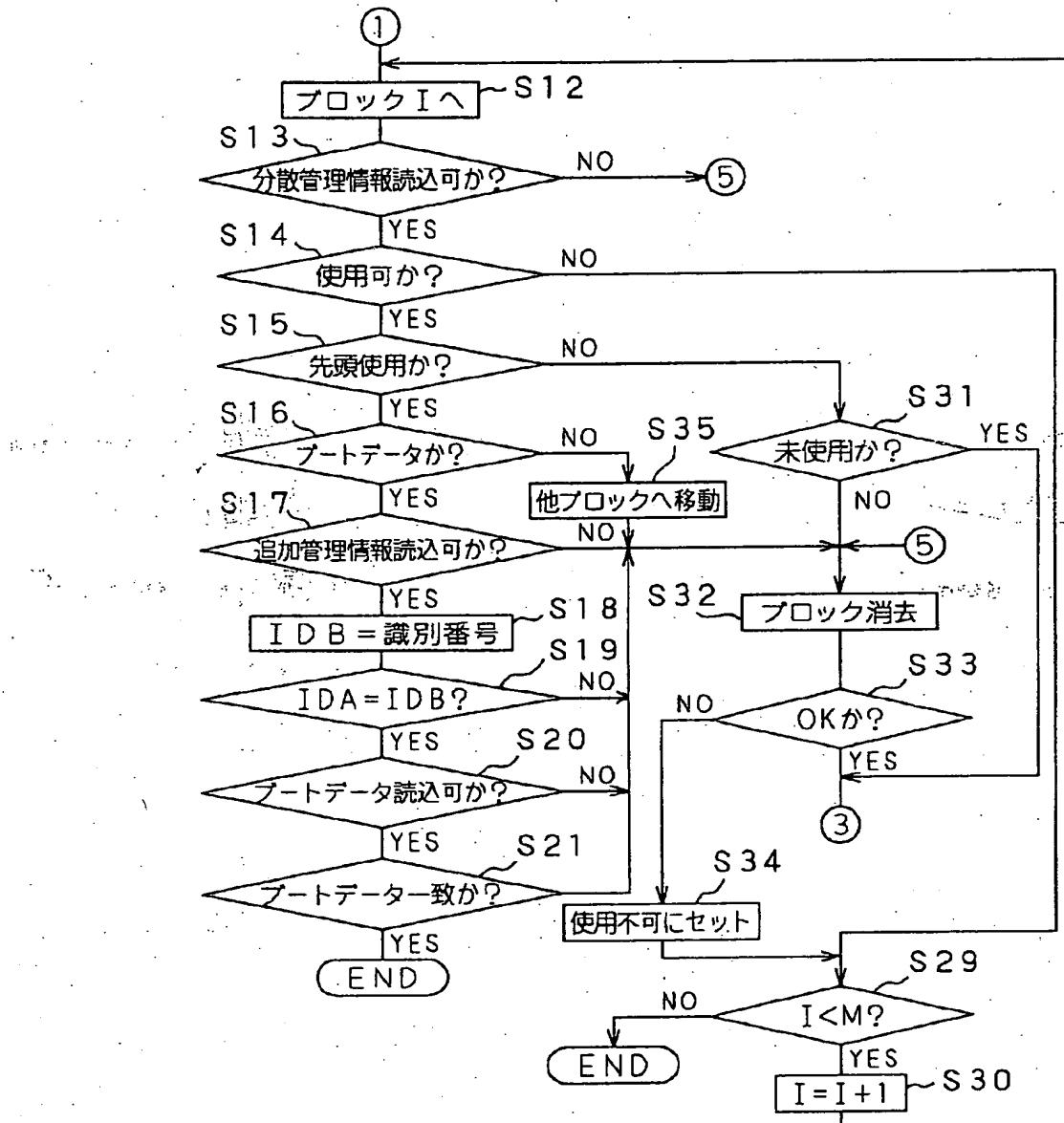


メモリカード起動時のブートブロック読み込みの手順

【図9】

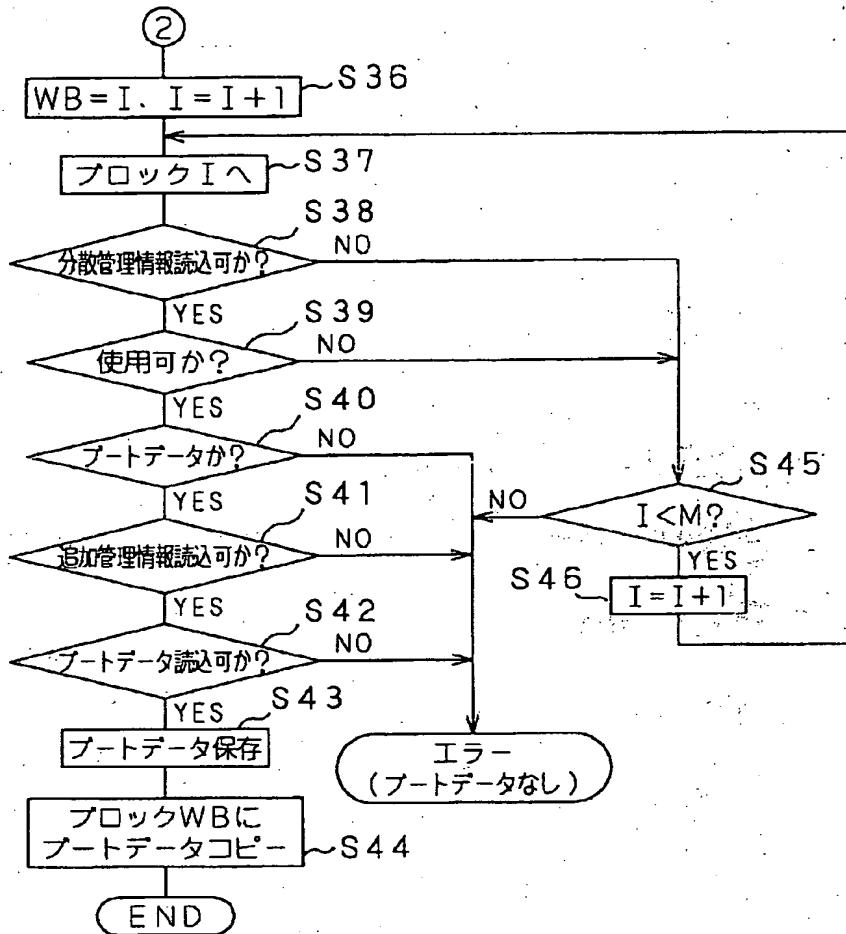


【図10】



メモリカード起動時のブートブロック読み込みの手順

【図11】



メモリカード起動時のブートブロック読み込みの手順

## 【手続補正書】

【提出日】平成11年1月26日

## 【手続補正1】

【補正対象書類名】明細書

【補正対象項目名】特許請求の範囲

【補正方法】変更

## 【補正内容】

## 【特許請求の範囲】

【請求項1】 データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置であって、ブートデータがそれぞれ格納された複数の異なるブロックを備えるとともに、ブートデータが格納された各ブロックに、それらのブロ

ックに格納されたブートデータの新旧を示す識別番号が格納される識別番号格納部を備えていることを特徴とする外部記憶装置。

【請求項2】 データの消去を所定のブロック単位で行う外部記憶装置に、当該外部記憶装置の起動時に最初に読み出されるブートデータを格納する際に、複数の異なるブロックにブートデータをそれぞれ格納するとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号を格納することを特徴とするデータ処理方法。

【請求項3】 上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納

された各ブートデータのうち、最新のブートデータを読み出して、当該ブートデータを用いて外部記憶装置を起動することを特徴とする請求項2記載のデータ処理方法。

【請求項4】 上記外部記憶装置を起動する際に、上記識別番号に基づいて、上記複数の異なるブロックに格納された各ブートデータの新旧を判別し、古いブートデータがある場合には当該ブートデータを最新のブートデータに書き換えることを特徴とする請求項2記載のデータ処理方法。

【手続補正2】

【補正対象書類名】明細書

【補正対象項目名】0010

【補正方法】変更

【補正内容】

【0010】

【課題を解決するための手段】本発明に係る外部記憶装置は、データの消去を所定のブロック単位で行うとともに、起動時に最初に読み出されるブートデータを格納するブロックを有する外部記憶装置である。そして、ブートデータがそれぞれ格納された複数の異なるブロックを備えるとともに、ブートデータが格納された各ブロックに、それらのブロックに格納されたブートデータの新旧を示す識別番号が格納される識別番号格納部を備えていることを特徴としている。